Available online at www.sciencedirect.com

ScienceDirect

Procedia Engineering 64 (2013) 844 – 853

Procedia Engineering

www.elsevier.com/locate/procedia

International Conference On DESIGN AND MANUFACTURING, IConDM 2013

"Heuristic based task allocation algorithm for multiple robots using agents"

"Thareswari Nagarajan[a], Asokan Thondiyath[b]"

"[a] Robotics Laboratory, Department of Engineering Design, Indian Institute of Technology Madras, Chennai 600036, India"
"[b] Robotics Laboratory, Department of Engineering Design, Indian Institute of Technology Madras, Chennai 600036, India"

Abstract

Task allocation plays an important role in achieving high utilization of robots in multiple robots system. Existing task allocation schemes do not consider the dynamic and unpredictable nature of the environment and try to maximize the robot synergism in order to achieve various objectives such as minimization of turnaround time, make span, and cost. Here the methodology for task allocation using a set of static and mobile agents controlling a pool of multiple robots is presented and the objective is to minimize the turnaround time. The agents are capable of autonomous decision making and thus make autonomous task allocation decisions based on the changing status of the robot system. Also, the agents are used for delivering the task and collecting the results in two separate trips thus avoiding the waiting time, which in turn helps to minimize the turnaround time. The performance of the algorithm is assessed by comparing with other existing task allocation method. The numerical simulation results manifest that the proposed algorithm performs the best under different problem scales and connectivity.

Keywords: "Multiple robot system, mobile agents, turn around time"

1. Introduction

Multiple robot systems [1] provide enhanced capabilities in applications like natural calamity, hostile terrain etc. which are too dangerous, expensive, and difficult for the humans to perform. Manufacturing automation is another potential application area for multiple robot system as they provide increased reliability due to redundant resources provided by the heterogeneous nature of the multiple robots [2]. However, expanding from a single robot entity into multiple entities makes the system more complex. The complexity comes in the form of intelligent coordination and execution of the tasks by the multiple robots in the system.

doi:10.1016/j.proeng.2013.09.160

Task allocation is one of the major areas of interest in multiple robot system as it addresses the issue of finding a task-robot mapping that achieves some system objectives such as improving turnaround time, cost, global utility, or distribution of resources [3, 4]. The existing task allocation approaches available in the literature [8-13] do not consider the dynamism within the robot system as well as the unpredictable nature of the application. Since the majority of the applications using multiple robots system are happening in the dynamic and unpredictable environment, this type of allocation has a major drawback when applied for multiple robot system due to their dynamism. This dynamism allows some robots to leave the system or change their locations in the system. Also, some new robot may be added to the system that may provide better performance as needed by the applications due to their better capability.

But these new robots will not be useful in static allocation situations. The unpredictable nature of the environment may make the physical motion of the robot and thus the task execution difficult. In such a situation, task migration is the only effective way to guarantee that the tasks are completed. Agent migration is much easier than the physical motion of the robots and is commonly adopted in most of the mobile agent-based methods [5,6,7]. These methods assume that there are some mobile agents that will carry task for their allocated resources and will wait for results before going to the next resource. This waiting will cause some delay leading to the increase in execution time. To avoid this situation, we propose a task allocation algorithm based on heuristics, which uses mobile and static agents to minimize the turnaround time.

Nomenclature

TR	turnaround time matrix
C_{ij}	turnaround time
S_j	processing speed
R	robot
T	task

2. Problem Formulation

A multiple robot task allocation problem with “m” robots and “n” tasks is considered here. The robots present in the system are assumed to be heterogeneous. The task allocation approach uses the turnaround time as the objective for mapping the robots and the tasks. The turnaround time matrix denoted by TR is prepared after collecting the details of available robots (more details on building this matrix is given in Sec.3). Each entry in the matrix represents the turnaround time of robot (Rj, j=1:m) for the corresponding task (Ti, i=1:n). The matrix TR will be of the form as shown in equation 1:

TR	R_1	.	.	.	.	R_m
T_1	C_{11}	.	.	.	.	C_{1m}
.	.	.		.	.	.
	.	.	.	.	.	.
.	.		.	.	.	.
	.	.		.	.	
T_n	C_{n1}	.	.	.	.	C_{nm}

(1)

where C_{ij} is the turnaround time for task *i* when executed on robot *j*. The turnaround time is the summation of the transmission time from the task assigner to the robot on which the task will be executed (Tr), the task execution time on that robot (Te), and the transmission time for task results from the robot to the task assigner (Tt).

The turnaround time of a robot '*j*' for a task '*i*' is given as:

$$Cij = Trij + Teij + Ttij \quad (2)$$

Tr_{ij} is calculated as

$$Trij = Li / Mj \quad (3)$$

where L_i is the size of a given task *i* and M_j is the bandwidth between the task assigner and the robot *j* on which the task i can be executed.

Te_{ij} is obtained as

$$Teij = Pi / Sj \quad (4)$$

where P_i is the execution time needed for the task i and S_j is the processing speed of the robot j.

The transmission time taken for transferring results from robot to the task assigner is defined by

$$Ttij = LR / Mj \quad (5)$$

where LR is the size of the task results obtained after executing task i and M_j is the bandwidth between the task assigner and the robot *j*. Each row in the TR matrix is sorted in an ascending order according to the turnaround time of each robot.

A task allocation algorithm assigns the tasks to the robots. The allocation algorithm tries to find a task-robot allocation that minimizes the overall turn around time i.e. $\sum Cij$. Based on this, the tasks are migrated to the selected robot and executed on it. This approach tries to remove the waiting time by considering the task delivery and result collection as two separate tasks. The agents travel once for delivering the task to the allotted robot and another time for collecting the results.

3. System Architecture

The components of the proposed allocation architecture are shown in Figure 1. It contains a user interface module, a robot attributes module, and a task assigner module. Through the user interface module, a user can submit missions or applications that have been divided into subtasks and each task needs a specific resource to be executed. The robot attributes module collects information about all robot resources from the multiple mobile robot system and stores it in the central resources database.

The attributes for each robot includes the robot speed, the robot type, the bandwidth between the task assigner and the other robots, the current capacity of the robot and the current availability of the robot. This module contains a sub module called updater. The main function of the updater is to update the database of robots with new information. The updater generates a mobile agent to collect robot information from the multiple mobile robots. The updater uses the incoming information to update the robot database.

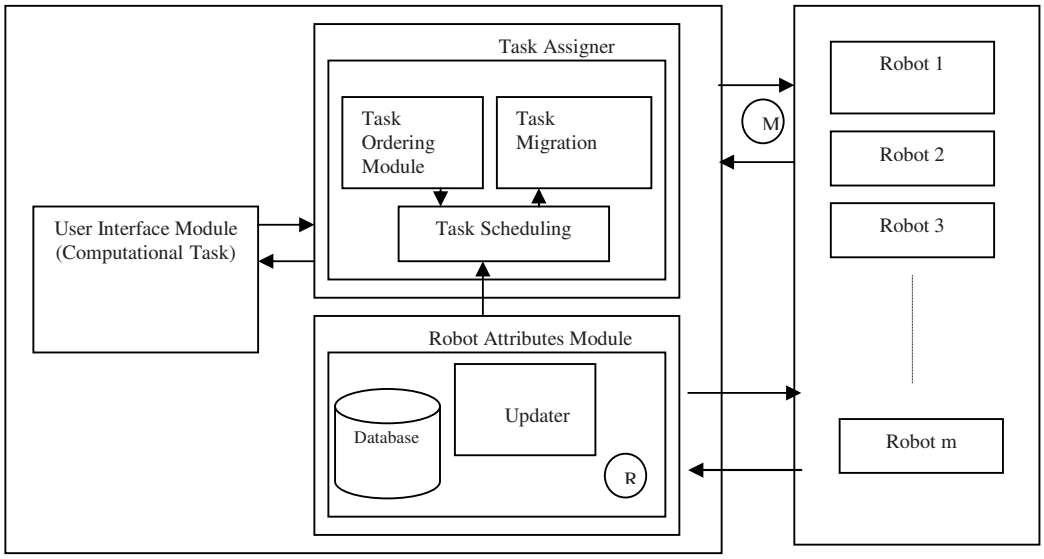

Fig. 1. Proposed system structure

The task assigner module is the heart of the proposed system. This module is responsible for assigning available robots of the system to the tasks. It selects the most appropriate robot to execute a task. The selection of the robots is based on the attributes retrieved from the robots database in the robot attributes module. The task assigner is divided into three sub modules as: task ordering, task scheduling and task migration. The task ordering module does the queuing of the tasks according to the priority. The task scheduling module is responsible for assigning available robots of the system to the tasks. It selects the most appropriate robots to execute a task. It has the list of suitable robots that can be used to execute the task. Then the task migration module dispatches the task to the robot that will execute it. It puts the task and the list of robots prepared by the task assigner module into a mobile agent and delivers that agent to the first robot in the list.

The system consists of a set of agents. These agents are distributed into the modules of the system. The robot attributes module has an agent called repository agent(R). This agent receives requests about robots information from the task assigner module. Then, it forms queries, sends it to the robot database and receives robot information from the repository. Finally, the agent sends the task assigner module the results of queries. The user interface module has interface agent(U) which is static and task assigner has two agents one for task ordering module(O) which is static and mobile agent called mobile task agent (M) for task migration.

4. Proposed Algorithm

The purpose of the proposed system is to assign tasks of the given applications to the most suitable robots in the system. The proposed algorithm for the system is given in the Table 1. The agent in the user interface module receives tasks for the application and sends it to the task assigner module. The static agent in the task ordering module receives the task attributes and determines the turnaround time matrix. The task migration module generates mobile task agents and these agents are sent to robot carrying tasks to be executed. Along with the task, mobile task agent must carry the list of robots. Each mobile task agent can carry all the tasks assigned to the robots in the same system. So, there is only one mobile agent for each system. This can preserve the bandwidth of the system and save communication overheads. Each mobile task agent has two processing steps in the system as

shown in figure 2 and 3. The first processing step is called the conveying. During this step, each mobile task agent will be directed to the first robot in the list of robots. If the first robot is unavailable due to some failure, the agent will be autonomously directed to the next robot in the list. This process will be repeated until the robot finds an available robot to execute the task. Then, the agent will deliver the task to that robot to execute it. After that, the agent considers the next task and assigns it to a robot using the same procedure.

Table.1 Proposed Algorithm

Algorithm
(1) Task assigner receives task attributes from the task ordering module and the robot attribute from the robot attributes module (2) From the attributes the turn around time matrix is formed (3) Task scheduling receives a list of tasks from the task ordering module and allocates the tasks accordingly (4) Then conveys schedule of the tasks to the task migration module (5) Task migration module generates mobile task agent (6) Task migration module puts on the mobile task agent with the list of task and robots for the given application or mission (7) Each mobile task agent do a conveying and collection process

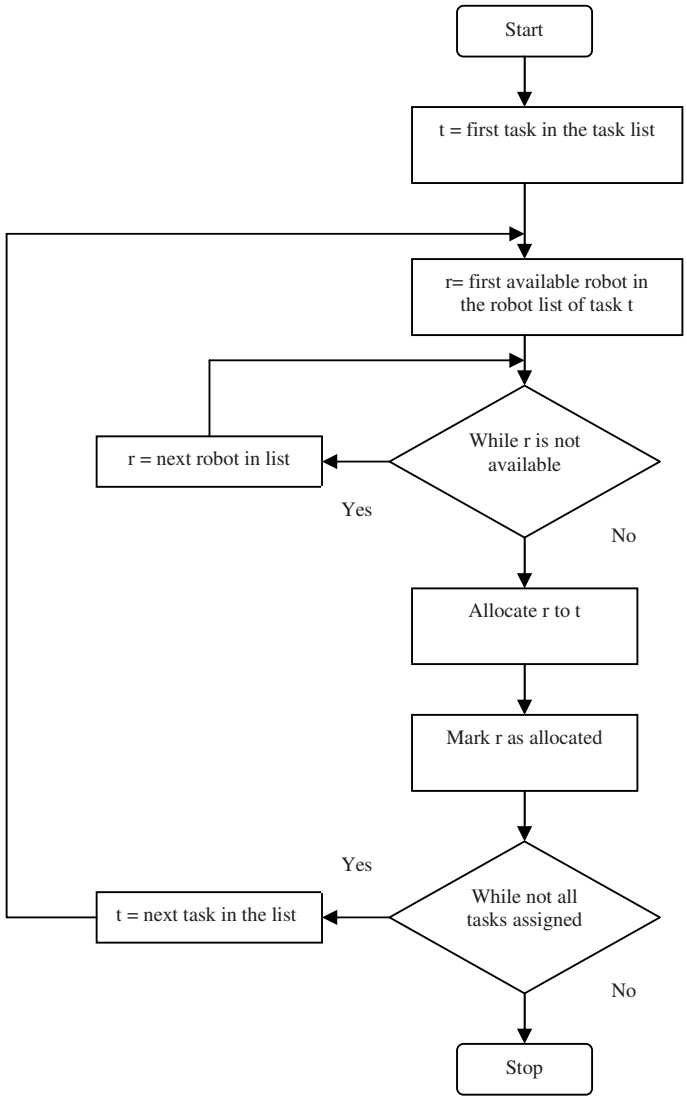

Fig. 2. Flow chart for the conveying process

In the second step, which is called the collection, the mobile task agent will start collecting results from the robots in the system. During this step, the agent will only pass on the allocated robots to collect results of tasks. After finishing the collection of results, the agent will return back to the task assigner module. The task assigner module can now convey results to the user interface module and thus the user can obtain results. Also, the mobile task agent makes a list of the unavailable robots during its processing. At the end of the process, the mobile task agent will notify the robot attributes module with the list of unavailable robots to update its database.

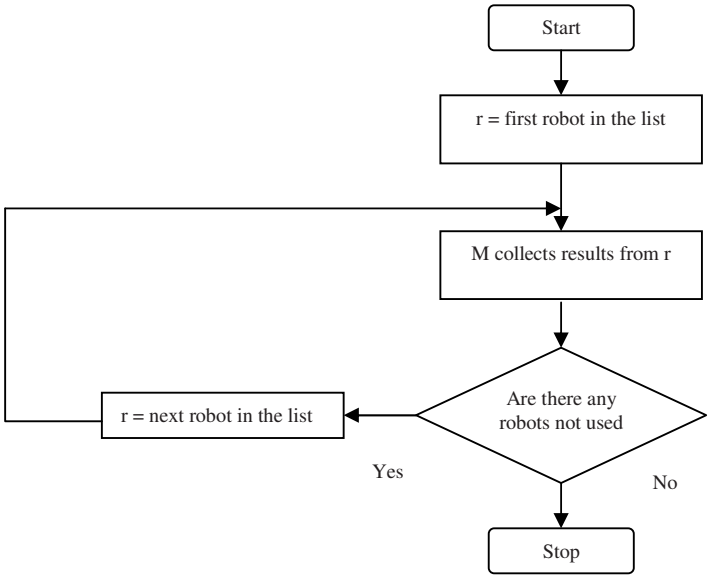

Fig. 3. Flow chart for the collection

5. Simulation

Numerical experiments are used to analyze the proposed algorithm and to study the effects of using it to reduce the turnaround time, and the influence of the number of tasks and robots on the performance. The performance of the algorithm will be compared with an auction method [2] in terms of turnaround time. The simulation environment is developed on Java platform that provides services for managing the various features of the application execution. The maximum numbers of robots and tasks are 50 and 200 respectively. The size of the tasks ranges from 1kb to 7mb. The simulation starts with the process of collecting the robots attributes from the robots present in the system by the mobile agents as shown in the figure 4.

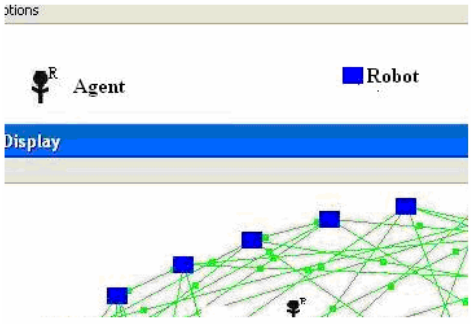

Fig. 4. Screen shot for agent collecting the robot attributes

Then the turnaround time is calculated for the given robots as shown in figure 5 for forming the turnaround matrix, representing the turnaround time for the tasks in every robot. For example suppose we have five robots and ten tasks then the turnaround time is given as shown in table 2 .Then it is used for mapping the tasks to the robots as shown the table 3.

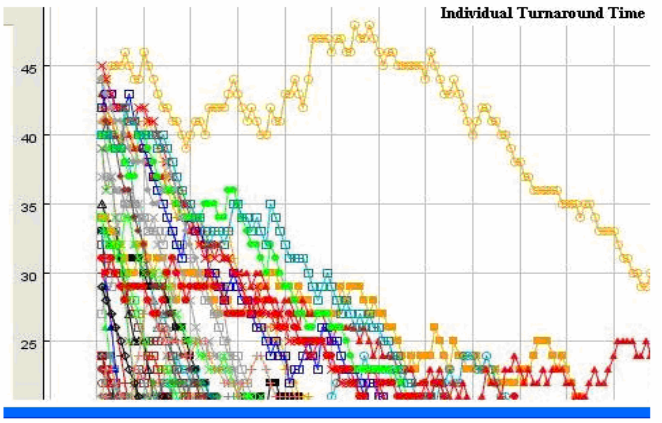

Fig .5. Screen shot for individual turnaround time

Once the mapping is done, the algorithm dispatches the list through the mobile task agent to the robots. The available robots are assigned the tasks in the conveying process and the results from the robots are collected by the collection process as shown in the figure 6. The table 4 shows the waiting time for the proposed method and the auction method for the turnaround time matrix given in table 2. The generation of agents is directly proportional to the various domains the tasks belong to.

Table.2.Trunaround time matrix

TR	R1	R2	R3	R4	R5
T1	475	255	433	386	387
T2	413	302	495	452	413
T3	313	414	479	432	492
T4	455	444	446	212	465
T5	492	475	202	472	404
T6	438	500	264	438	464
T7	458	497	415	486	206
T8	474	426	421	204	426
T9	399	285	406	414	473
T10	259	473	445	412	499

Table.3.Robot task assignment

Robot	Assigned Task
R1	T3,T10
R2	T1,T2,T9
R3	T5,T6
R4	T4,T8
R5	T7

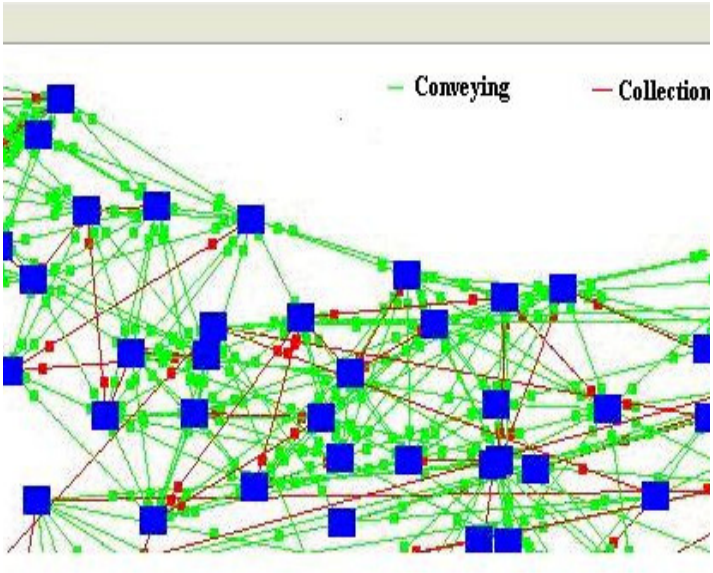

Fig.6.Screenshot for conveying and collection process

Table.4 Comparison for waiting time

Algorithm	Waiting time
Proposed Method	212.80 (usage of agents)
Auction Method	806.17

A comparison of the performance of the proposed algorithm and auction method with respect to no. of robots in the system is shown in figure 7. It can be seen that the proposed algorithm provides turnaround time better than the other algorithm. This is due to the extra time taken in the auction methods by the robots awaiting the results from the other robots. In the proposed system, by the usage of the mobile agent there is no need for the extra waiting time for the results. In both the methods, the turnaround time decreases as the number of robots increases.

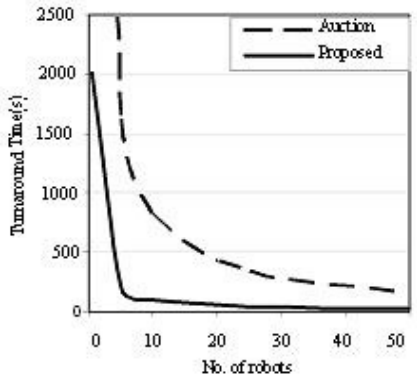

Fig.7. Comparison of proposed algorithm and auction method

Figure 8 shows the comparison for studying the effect of task size in the proposed system and the auction method. Here it is shown that, as the number of tasks increases the memory size also increases in both the methods. The rate of this increment is slower in case of the proposed system than in the case of the auction method. This is due to the fact that each mobile agent in the proposed system can carry many tasks to different robots. It means that the proposed system will remain effective when there is a significant increase in the number of robots and the number of tasks to be executed.

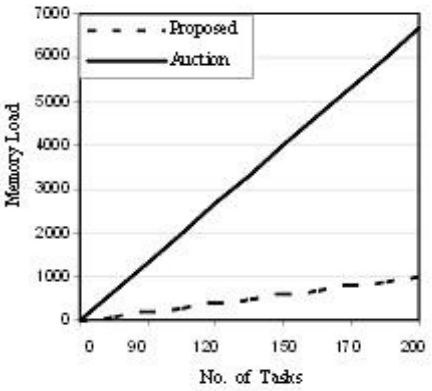

Fig.8.The effect of increase in number of tasks

6. Conclusions

In this paper, the task allocation algorithm for the multiple robots has been proposed using agents. The algorithm uses the mobile agents for achieving flexible, scalable and intelligent coordination system. The agents help in the multiple robots system for identifying the robot failure and also usage of the mobile agents helps in better execution of multiple robot system using less communication in terms of lower bandwidth. The proposed algorithm is compared with the auction methods and it is proven that the proposed system provides better scalability. Also, the proposed algorithm for using mobile agents is compared with the auction method, in terms of turnaround time. It is proven that the approach used in the proposed system provides better turnaround time than the other method.

References

[1] R. Zlot, A. Stentz., 2006. Market-based multirobot coordination for complex tasks. International Journal of Robotics Research 25(1), p.73-101.
[2] B.P. Gerkey, On multi-robot task allocation [D]. University of South California.
[3] R. Zlot, A. Stentz., 2005. "Complex task allocation for multiple robots", International Conference on Robotics and Automation. Spain, p.1515-1522.
[4] B.P.Gerkey , M.J.Mataric, 2004. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems, The International Journal of Robotics Research 23, p.939-954.
[5] Y. Y. Zhang, J. D. Liu., 2003. Mobile agent technology, Tsinghua University Press, Beijing.
[6] Y. Kambayashi, M. Takimoto., 2005. Higher-order mobile agents for controlling intelligent robots. International Journal of Intelligent Information Technologies,1(2),p.28–42.
[7] M. Takimoto, M. Mizuno, M. Kurio, and M. Kambayashi., 2007. Saving energy consumption of multi-robots using higher-order mobile agents . Lecture Notes in Artificial Intelligence 4496, Springer-Verlag, p.549–558.
[8] P.Svestka, M.H. Overmars., 1998. Coordinated Path Planning for Multiple Robots. Robotics and Autonomous Systems 23(4), p. 125-133.
[9] M.B. Dias and A. Stentz., 2000. "A free market architecture for distributed control of a multirobot system",6th International Conference on Intelligent Autonomous Systems. IAS-6, p. 115.
[10] S.S. C. Botelho and R. Alami., 1999. M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement, Robotics and Automation 2, p.1234-1239.
[11] A.Viguria, I.Maza, and A. Ollero., 2008. "S+T: An Algorithm for Distributed Multirobot Task Allocation based on Services for Improving Robot Cooperation" , The IEEE International Conference on Robotics and Automation(ICRA). Pasadena, EEUU.p 3163–3168.
[12] A.Viguria, A. Howard., 2009. Probabilistic Analysis of Market-based Algorithms for Initial Robotic Formations. The International Journal of Robotics Research, DOI: 10.1177/0278364909340333.
[13] N.Thareswari, T.Asokan., 2011. "An improved algorithm for constrained mutirobot task allocation in cooperative robot tasks", International Conference on Intelligent Robotics, Automations, telecommunications facilities and applications. Korea, p. 455-466.